Real-Time Imaging **4,** 55–65 (1998)

Real-Time Visual Sensing for Task Planning in a Field Navigation Vehicle

Some agricultural tasks consist of applying chemical treatment to the crops, but the products are applied throughout the field in most cases. In order to automate these tasks, and to apply the products more accurately, an autonomous vehicle can be used. The vehicle is intended to navigate autonomously through a field where plants are arranged in rows, following the crop rows. Its main sensor is a camera which takes perspective images of the area close to and in front of the vehicle. The vehicle is equipped with a treating device consisting of a bar with 30 spray nozzles. When the plants, which have been identified at some distance in front of the vehicle, reach the position of the treating device, the nozzles which are over a plant are switched on, thus applying the treatment. The typical vehicle speed is 1 m/s and the video frame rate is 10 Hz. A distributed system, consisting of several processors interconnected by serial links, is used, where each module accomplishes a different task: vehicle control, guidance information, image segmentation, map building and nozzle switching. The work explained in this paper is mainly focused on the latter parts of the system, map building and nozzle switching, and stresses the methods used to fit the time requirements. Real-time performance is achieved. Some results and time measurements are given.

J. M. Sanchiz[1], J. A. Marchant[2], F. Pla[1] and T. Hague[2]

[1]Department of Computer Science, University Jaume I, Campus de Penyeta Roja, 12701 Castellon, Spain, [2]Image Analysis and Control Group, Silsoe Research Institute, Wrest Park, Silsoe, Beds. MK45 4HS, UK

Introduction

Chemical treatments are widely used in crop protection tasks, yet the treatment is applied throughout the field in most cases, resulting in a waste of product. The automation of these tasks could be achieved by an autonomous crop-protection vehicle, designed to move along a field in which crops are arranged in rows, with a visual system for the steering and the identification of plants and weeds, and with a bar with nozzles to apply the treatment.

As the vehicle moves through the field, images are taken at a predetermined rate, they are segmented to separate plants, weeds and soil, and they are used to find the heading angle and the offset of the vehicle with respect to the crop rows, and to know which nozzles to switch on, and when. Such an autonomous vehicle (Figure 1) has been developed at Silsoe Research Institute, U.K., and this work is related to what can be considered the last stages of the overall system, receiving the segmented images, accumulating evidence of plants, weeds and soil, building a local map of the field, and exploring this map along the positions of the spray nozzles to switch on the correct ones, thus applying the treatment to the plants or to the weeds.

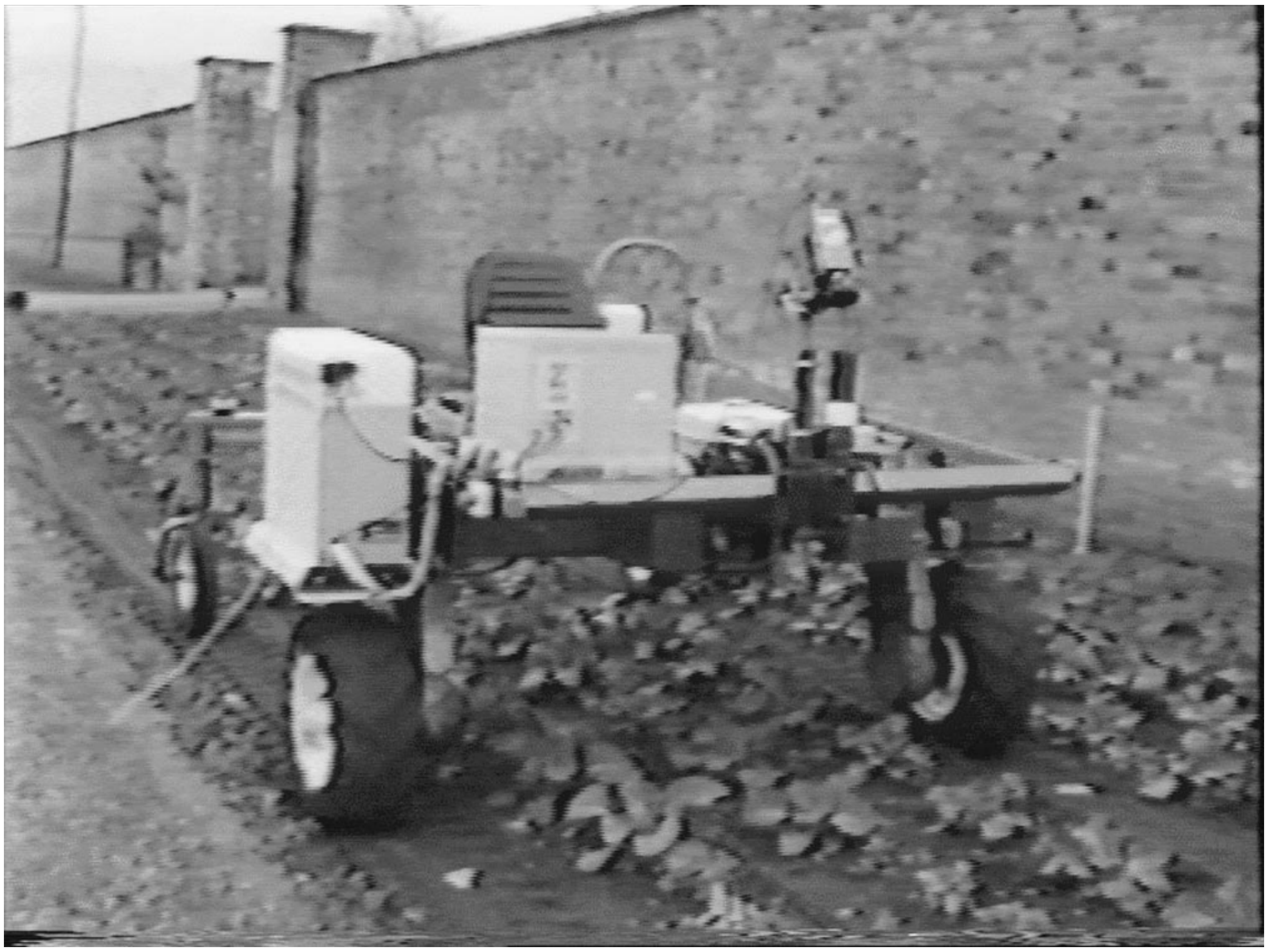


Figure 1. Autonomous vehicle in a field.

Images are taken at a rate of 10 Hz and the typical vehicle speed is 1 m/s. One of the main goals of the system is that it works in real time.

The general discrimination problem is a very difficult one and, in order to make some progress, we felt that a restriction to the working conditions of the system was necessary. Thus, the problem has been confined to a restricted, but important, horticultural situation: that of transplanted vegetable crops. In commercial practice the plants are grown to about 75 mm high in a greenhouse, then transplanted into an almost weed-free seedbed. Thus, the plants are nearly always bigger than any weeds that may grow. Also the plants are transplanted into a reasonably regular pattern. Our target treatment is weeding, which is done at intervals throughout the early life of the plants. After the crop has grown to a reasonably mature stage, competition between plants means weeding is not so necessary. Thus we have confined ourselves further to situations where the plants remain as spaced blobs in the images.

Some vehicle navigation applications have been reported in recent years, most of them related to road following in outdoor environments [1–5], and others related to indoor and structured scenes, where navigation is made inside a building and the main features to focus on are edges of corridors, walls, doors, etc. [6, 7]. In the present application the only structured features are the rows in which plants are aligned, and this information is used for the steering of the vehicle, together with information from other sensors such as an electronic compass and wheel shaft encoders. By knowing the vehicle frame-to-frame motion, and using the calibration data from the camera, a local map in a zone around the camera, which includes the position of the nozzle bar, can be recovered. Other works related to surface reconstruction [8, 9] involve the tracking of

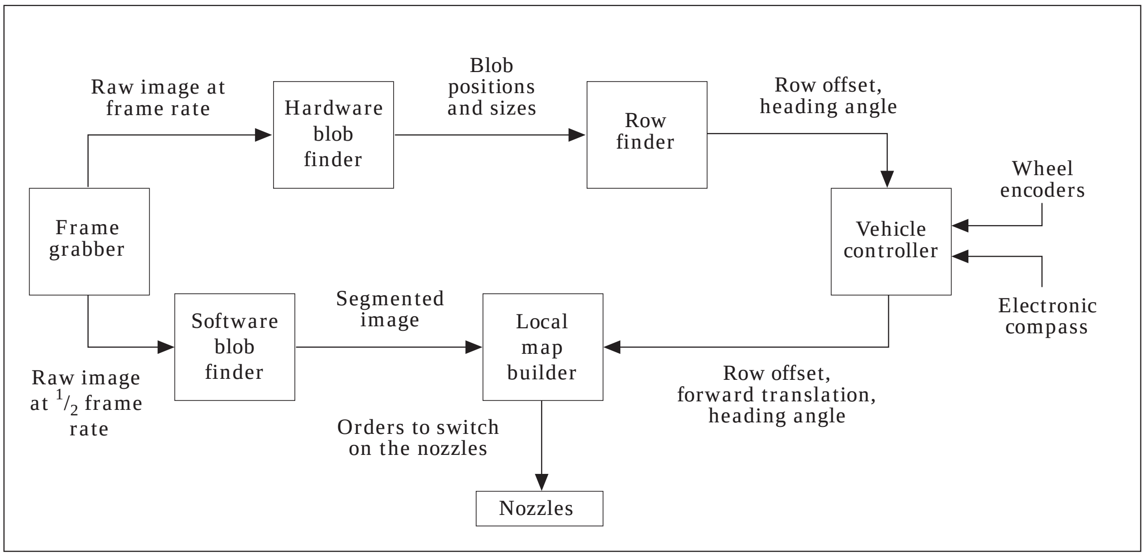


Figure 2. System modules and relations.

feature points at different heights from the ground. In our case the objective is to spray the plants in real time, which constrains our surface reconstruction method to place the field of view on the ground plane at each frame. Consecutive images overlap in an area of the field of view, and this overlapping is used to filter some segmentation errors, like small blobs that can be incorrectly classified as plants in one image, and then classified as weeds in the next one. The decision to switch on some nozzles is taken by exploring this local map along the position of the nozzle bar. The system latency can be corrected by moving the position of the bar forward according to the vehicle speed.

System Overview

The relationships between the different modules of the system, and the information that is being passed between modules, can be seen in Figure 2. A short description of each module follows:

– Frame Grabber. Its purpose is to acquire raw images. Images are taken with an infrared filter that enhances the contrast between vegetation and soil. The frame rate is 10 Hz.
– Hardware Blob Finder (chaincoder). A dedicated hardware component [10] that segements the image with a fixed gray-level threshold, to classify the pixels in two classes: vegetation (plants and weeds) and soil. It then finds the center and size (area) of each blob of class vegetation. This information is used by the Row Finder.
– Row Finder. A software module that computes the position of the three rows that best match the information of the blobs received from the blob finder. The module finds the offset of the camera with respect to the central row, and the heading angle. This is done by using a special Hough Transform technique [11], where the offset and the heading angle form a two-dimensional parameter space [12].
– Software Segmentation. Every other frame is input to the segementation module to achieve real-time performance. Segmenting the images and building the local map are the most time-consuming modules, and cannot be done at the 10 Hz grabbing rate. Using alternate frames allows 200 ms to process each image. This is a two-threshold segmentation, obtaining three classes of pixels: soil, weeds and plants, [13]. The thresholds are computed automatically from the histogram. The reason not to use the hardware blob finder here is that it only applies a simple thresholding; good enough for the row finder, but not for plants/weeds/soil segmentation, where more extensive processing is needed.
– Vehicle Controller. Receives information from sensors, the offset and the heading angle from the Row Finder, orientation information from an electronic compass, and wheel positions from shaft encoders. It controls the steering and forward speed of the vehicle, and sends optimally estimated measurements of the forward translation, transverse offset and heading angle to the Local Map Builder.

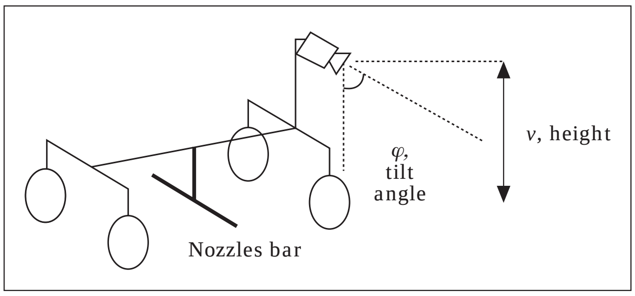


Figure 3. Geometry of the sensor (camera) and the treating device.

– Local Map Builder. It accumulates evidence at each pixel of observations of plants, weeds and soil. It places the sequence of images on a local map around the vehicle, displacing this map as the vehicle moves. It places the nozzle bar on the local map and explores the positions of the nozzles, to switch on those ones that are over a plant. As its input comes from the software segmentation module, alternate frames are used to build the map, giving a speed requirement of 5 Hz.

Geometry of the Problem

Figure 3 shows the geometry of the main sensor of the system, the camera, and of the actuator, the nozzle bar. The camera has a perspective view of the nearby part of the field: a typical image can be seen in Figure 4(a) and the corresponding segmented image in Figure 4(b), where the three classes, plants, weeds and soil, can be easily identified. The row structure found by the row finder is shown in Figure 4(c). The tilt angle, φ, and the camera height, v, are fixed, and their values are obtained from a previous calibration [14]. A side view of the camera and the ground can be seen in Figure 5(a), and an overhead view in Figure 5(b).

The swing angle of the camera is considered to be zero, and the tilt angle is considered constant. Only the heading angle (pan angle) is assumed to vary as the vehicle moves, and its value, together with the translation vector, is recovered from the visual data and the other sensors of the system, and optimally estimated in the Vehicle Controller module. Small stones or holes in the path of the vehicle may cause it to tilt or roll very slightly. So far we have achieved consistent row following; however, this could cause errors in the local map which do not affect row following. This small tilt and roll, if any, is very difficult to recover from the visual data, but we have recovered it from accelerometers, thus allowing the corrections of these errors in future work. Nevertheless, the system is designed to work in quite a controlled environment, where the seedbed, as mentioned before, has been carefully prepared, and so is expected to be stone and hole-free.

Given a point in the image (x_{comp}, y_{comp}), its world coordinates (x_w, z_w) can be found knowing φ (tilt angle), v (height), Ψ (heading angle), t_x (absolute x-translation, offset) and dt_z (z-translation relative to the previous image). The world coordinates are defined to be aligned with the crop rows; the z_w axis is along the row direction, the x_w axis is perpendicular, and the origin is the projection of the camera axis on the ground at the first image of the sequence.

The computations on the world coordinates of a pixel, given its computer coordinates, are as follows:

- from computer coordinates (x_{comp}, y_{comp}) to image coordinates (x_{image}, y_{image}).

$$\begin{pmatrix} x_{image} \\ y_{image} \end{pmatrix} = \begin{pmatrix} dx \dfrac{(x_{comp} - c_x)}{\mathrm{sx}} \\ \mathrm{dy}(y_{comp} - c_y) \end{pmatrix} \qquad (1)$$

(c_x, c_y) being the center of image coordinates. Since the size of the images is 256×256 pixels, using a pin-hole model of the camera, the center is placed at (128,128);

dx, dy are the scale of image sensor, in mm/per pixel;
sx is the aspect ratio;
dx, dy and sx are obtained from the camera calibration [14]; no radial distortion is assumed.
The values used are $dx = 0.05$, $dy = 0.05$, $sx = 0.71$.

- placing a point in the image (x_{image}, y_{image}) on the ground plane, in camera coordinates (x_1, y_1, z_1).
The definition of the camera coordinates is:

x_1 axis is the same as x_{image} axis;
y_1 axis is the same as y_{image} axis;
z_1 axis is aligned with the optical axis
the center of coordinates is the center of the pin-hole model;

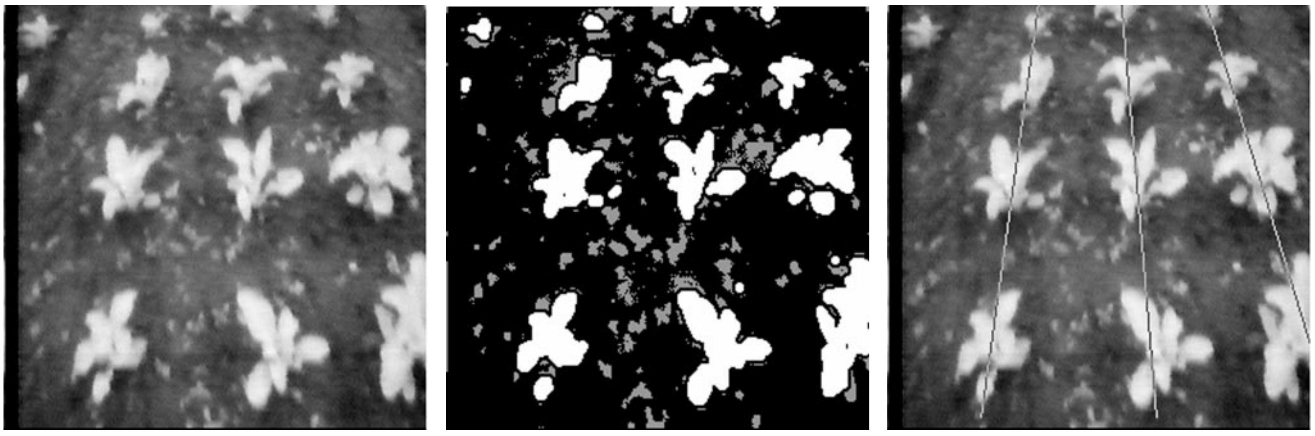


Figure 4. (a) Sample image. (b) Segmented image. (c) Row structure found by the Row Finder.

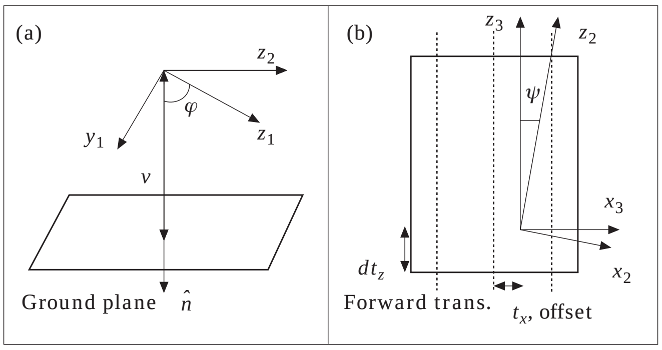


Figure 5. (a) Camera coordinates, side view. (b) Overhead view.

Equation of the ground plane in camera coordinates: $\sin(\varphi)\ y_1 + \cos(\varphi)\ z_1 - v = 0$;
a pixel with coordinates (x_{image}, y_{image}) lies on the line $x/x_{\text{image}} = y/y_{\text{image}} = z/f$;
the intersection of this line with the ground plane is:

$$\begin{pmatrix} x_1 \\ y_1 \\ z_1 \end{pmatrix} = \begin{pmatrix} z_1 \dfrac{x_{\text{image}}}{f} \\ z_1 \dfrac{y_{\text{image}}}{f} \\ \dfrac{v}{y_{\text{image}} \dfrac{\sin\varphi}{f\varphi} + \cos\varphi} \end{pmatrix}$$

- finding the ground coordinates without row alignment (x_2, y_2, z_2).
  Assuming there is no roll angle, only a rotation around the x_1 axis of (90-φ) degrees has to be applied:
  Coordinates (x_2, y_2, z_2) are defined as follows:
  x_2-axis is the same as x_1-axis;
  y_2-axis is perpendicular to the ground plane;
  z_2-axis is parallel to the ground plane, it is the projection of the optical axis on the ground plane.

$$\begin{pmatrix} x_2 \\ y_2 \\ z_2 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \sin\varphi & \cos\varphi \\ 0 & \sin\varphi & \cos\varphi \end{pmatrix} \begin{pmatrix} x_1 \\ y_1 \\ y_1 \end{pmatrix} =$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & \sin\varphi & \cos\varphi \\ 0 & -\cos\varphi & \sin\varphi \end{pmatrix} \begin{pmatrix} x_1 \\ \dfrac{v - z_1\cos\varphi}{\sin\varphi} \\ z_1 \end{pmatrix} =$$

$$\begin{pmatrix} x_1 \\ \sin\varphi \dfrac{v - z_1\cos\varphi}{\sin\varphi} + z_1\cos\varphi \\ -\cos\varphi \dfrac{v - z_1\cos\varphi}{\sin\varphi} + z_1\sin\varphi \end{pmatrix} = \begin{pmatrix} x_1 \\ v \\ \dfrac{z_1 - v\cos\varphi}{\sin\varphi} \end{pmatrix} \tag{3}$$

y_2 always equals v as expected.

- finding the ground coordinates with row alignment (x_3, y_3, z_3).

A rotation around the y_2 axis of Ψ has to be made.

$$\begin{pmatrix} x_3 \\ y_3 \\ z_3 \end{pmatrix} = \begin{pmatrix} \cos\Psi & 0 & \sin\Psi \\ 0 & 1 & 0 \\ -\sin\Psi & 0 & \cos\Psi \end{pmatrix} \begin{pmatrix} x_2 \\ y_2 \\ y_2 \end{pmatrix} = \begin{pmatrix} x_2\cos\Psi + z_2\sin\Psi \\ y_2 \\ -x_2\sin\Psi + z_2\cos\Psi \end{pmatrix} \tag{4}$$

- finding the world coordinates (x_w, z_w) (with camera translation).

World coordinates are defined over the ground plane, so it is a two-dimensional system of coordinates.

$$\begin{pmatrix} x_w \\ z_w \end{pmatrix} = \begin{pmatrix} x_3 + t_x \\ z_3 + t_z \end{pmatrix} \quad (5)$$

where $t_z(\text{image } k) = t_z(\text{image } k\text{-}1) + dt_z$.

Building the Local Map

As mentioned before, a pixel in computer coordinates can be mapped on to the ground plane in world coordinates, but this is quite a time-consuming operation (nine multiplications/divisions and seven additions/subtractions), and it should be done with every pixel. But a real-time implementation is required, subject to these restrictions:

– since the map builder receives data from the software segmentation module, every image has to be processed at a rate of 5 Hz; that is, a processing time of 200 msec;
– the module has to run on the available hardware, a T800 transputer at 25 MHz.

Hence some decisions have been taken to speed up the process as much as possible:

– Since the images show a perspective view of the field, the lower half image shows much more detail than the upper half. Objects in the upper half are much further from the vehicle but will be seen nearer in the subsequent images. Hence we decide to process only the lower half, which means processing $128 \times 256 = 32\,768$ pixels per image.
– only the world coordinates of the four corners of the lower half image, which is going to be transferred to the local map, are computed, and the common part of the computations, until the heading angle has to be used, is obtained from a look-up table. This look-up table, containing the computations that only depend on the calibration data (including camera height and tilt angle), is built up in a boot-strap phase, and includes Eqns (1), (2), and (3).
– Figure 6 shows the outline of a single image, defined by its four corners, transferred to the local map. Knowing the corner positions we can visit each pixel of the field of view by computing the line from corner 1 to 3, and for each pixel on this line we can compute a line parallel to the one from corner 1 to 2. This defines a pixel visiting order. With this order we access another look-up table, where the corresponding computer coordinates of the pixel are directly found. This look-up table, which includes equations (1)–(5), is also built up in the bootstrap phase by placing a field of view at heading angle and forward translation zero, visiting every pixel in the field of view in the predefined order, and computing their computer coordinates. The displacements of the lines parallel to the one corner 1 to 2 need to be computed just once, since all these lines are parallel.
– visit alternate lines and/or alternate pixels on each line if the process does not fit the time requirements. In that case write in the non-visited pixels the same information as in the nearest visited pixel.

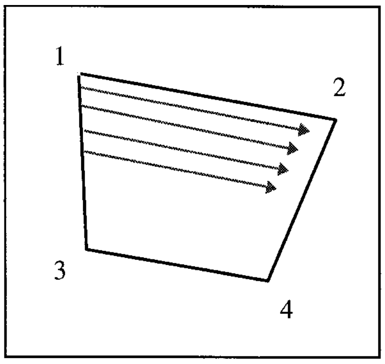


Figure 6. Field of view and pixel visiting order.

Accumulating Evidence of Plants and Weeds

There is some overlapping between subsequent images, but the overlapping regions may not be exactly the same. Some pixels assigned to a certain class, plant, weed or soil, can be assigned to a different class in the subsequent image. This is mainly due to:

– a different point of view from the camera. While the vehicle moves, plants and weeds approach to the camera, and they are 3D objects.
– segmentation errors. The segmentation module enhances the contrast between plants and weeds sufficiently to differentiate them using two different thresholds, but sometimes small areas of weeds or plants are misclassified. The result is that small blobs of class plants are obtained, when they are really part of a weed. These small blobs may disappear in the subsequent image, especially when the plants and weeds are approaching the camera. Real plants have a bigger size than these small blobs so a size filter could be used to discard them, but we have not been able to do this at the required rate.

Thus, for each pixel we accumulate evidence of plants, weeds or soil by counting the number of

Table 1. Processing times for different resolutions

Resolution	Time
8 mm/pixel. 1×1 resolution (visiting each line and each pixel on that line)	580 ms
8 mm/pixel. 3×1 resolution (visiting every third line and each pixel)	203 ms
8 mm/pixel. 2×2 resolution (visiting every other line and every other pixel)	201 ms
8 mm/pixel. 3×2 resolution (visiting every line of three and every other pixel)	168 ms
8 mm/pixel. 4×2 resolution (visiting every fourth line and every other pixel)	150 ms

observations of each class, and we decide which class is assigned by using a majority voting scheme. Each pixel is stored in a byte: three bits are assigned to the plant counter, another three to the weed counter, and two to the soil counter. Knowing the value of these counters we decide if the pixel is of class plant, weed or soil, by finding the counter with the biggest value. In the case of a draw the priority is weed, then plant and finally soil. If all three counters are zero the pixel is assigned to a special class, 'unknown'.

Switching on the Nozzles

The nozzles are arranged in a 1500 mm wide bar, at a separation of 50 mm. The line in the map representing the nozzle bar is explored at each nozzle position. Depending on what the target is, plants or weeds, a nozzle is switched on if a segment of the corresponding class, centered at the nozzle position, and bigger than a given threshold, is found. The threshold used for the segment's length depends on the map scale factor, and should represent a real length of about 50 mm, the separation between nozzles.

The time to process an image to build the local map is independent of the time of operation of the nozzles. The sampling rate is unimportant as long as we build a complete map of the local area, and the map segments join together with no gaps (or overlap, to give a factor of safety). The field of view is important in this respect, as well as the sampling rate. We have chosen our variables so that we observe each point in the field of view about three times, so we have sufficient overlap. In fact we use multiple views to improve the reliability of the segmentation. The operation time of the nozzles, and the time of flight of the spray drops, must of course be considered when deciding exactly when to turn the nozzles on or off; this has been taken into account to choose solenoids and nozzle height. The position of the nozzle bar has to be explored at a certain distance in front of where the bar is really in the map, thus taking into account the time of operation and the vehicle speed. Then each nozzle has to be active for no more than 200 ms, when it will receive a new order, but certainly the nozzle activation and spray flight latency do not affect the computing time, or rate requirements, of the map builder module.

Off-line Results

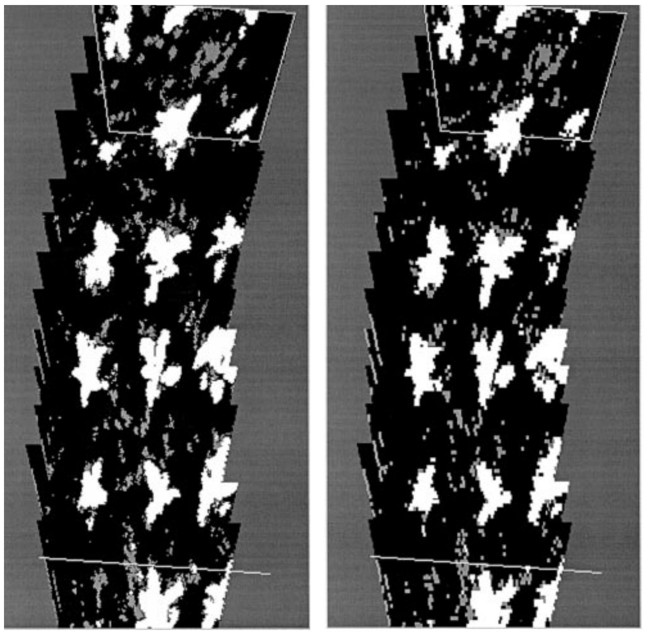
A first experiment was carried out to measure the timing of the process in the laboratory. Several sequences of 30 images were taken from a camera mounted on a manually-driven vehicle, running at a speed of 1 m/s and at a frame rate of 5 Hz. The software segmentation and the row finding were applied to every image, thus finding an approximate value for the heading angle and the row offset (transverse translation), the forward displacement was estimated by matching features manually in every pair of consecutive images. For alternate frames, i.e. at a rate of 5 Hz, this information was input to the map builder. The computing time was measured for different scales and resolutions of the map; as mentioned before, the limit is 200 ms per image. The camera geometry for this test was $v = 1200$ mm, $\varphi = 66°$, $f = 40$ mm. For this geometry, the maximum scale is 8 mm/pixel: a bigger scale results in a field of view too big to fit in the map. An image of 256×512 pixels was used to store the map, giving a real size of 2048×4096 mm for this scale. This process was run on a T800 transputer at 25 MHz; the processing time for different resolutions can be seen in Table 1.

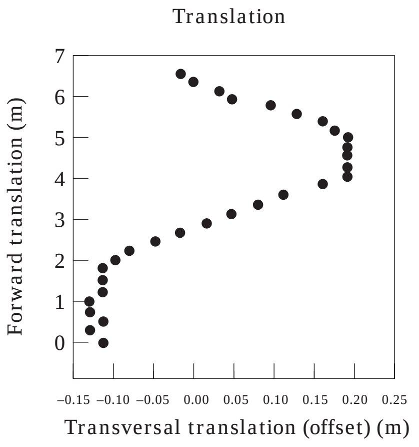
Figure 7 shows the recovered local map for one sequence, where a decision to classify each pixel as plant, weed or soil, using the majority voting scheme, has been made. Figure 7(a) shows the map built at 1×1 resolution and 8 mm/pixel; Figure 7(b) shows it at the same scale, but at 3×2 resolution, which fits the time requirements while still being satisfactory for the purpose of the work, deciding whether a nozzle is over a plant or a weed. The timing can still be improved further by reducing the scale of the map. The recovered trajectory for this sequence can be seen in Figure 7(c). The input data for the mapper in this test was generated off-line from the row finder, without any filtering by the controller. As this data comes from a Hough space, it is coarsely quantized and cannot be taken into account for accuracy measurements. The real accuracy of the system will be seen in the real-time implementation, when all modules are working.

Real-Time Experiments in a Simulated Field

Another experiment consisted of testing the whole system on the autonomous vehicle in a simulated field at full size, the plants were represented by blobs painted on the ground (Figure 8). Here the leading angle, row offset and forward translation were estimated automatically from the row finder and the vehicle controller modules. The camera geometry was $v = 1400$ nm, $\varphi = 30°$, f = 15.5 mm and the row spacing was 500 mm. The scale used to build the local map was 18 mm/pixel, thus representing a map of 9216 × 4608 mm. The field of view was smaller than that in the recorded images, which resulted in a decrease of processing time per image. Several tests were made with the vehicle travelling a distance about 8 m at a speed of 1 m/s, steering itself automatically to follow the simulated crop rows, and aligning the center of the vehicle with the central row. Using a 1 × 1 resolution, the processing time was close to 150 ms per image, thus fitting the specifications.

Figure 9 shows two maps and vehicle trajectories corresponding to two of the tests. In order to check the quality of the recovered maps, some measurements were made on them: for this purpose, the maps were stored in files after each test. One line was fitted to every row, using the centers of the blobs as points whose distance to the line was minized; these lines can also be seen in Figure 9. The measurements performed on these lines were: mean square error of the fit; parallelism between lines (rows are parallel in the real world); and separation between lines, which should match the real world separation between rows. The mean square error of the fit for the lines shown was 12.07, 14.61, 16.27, 8.50, 6.11 and 13.91 mm. The angle between pairs of consecutive lines was 0.0276°, 0.0499°, 0.0066° and 0.0126° (very close to zero, as it should be in the real world), and the separation between consecutive lines, measured at the center of the map, was 517.26, 505.90, 515.64 and 513.53 mm (very close to the real world separation between rows, 500 mm). So we can say that the local map has been recovered quite accurately for the purpose of the work. In fact only local accuracy is needed, since just the area between the actual field of view and the position of the nozzle bar has to be stored, and when the bar has passed over an image it can be forgotten as it is no longer needed for the system.


(a) (b)



(c)

Figure 7. (a) Local map at a scale of 8 mm/pixel and 1 × 1 resolution. (b) Local map at a scale of 8 mm/pixel and 3 × 2 resolution. (c) Recovered trajectory.

Conclusions

A real-time image processing application has been presented. Its purpose is to build a local map of the ground between the field of view and the treating

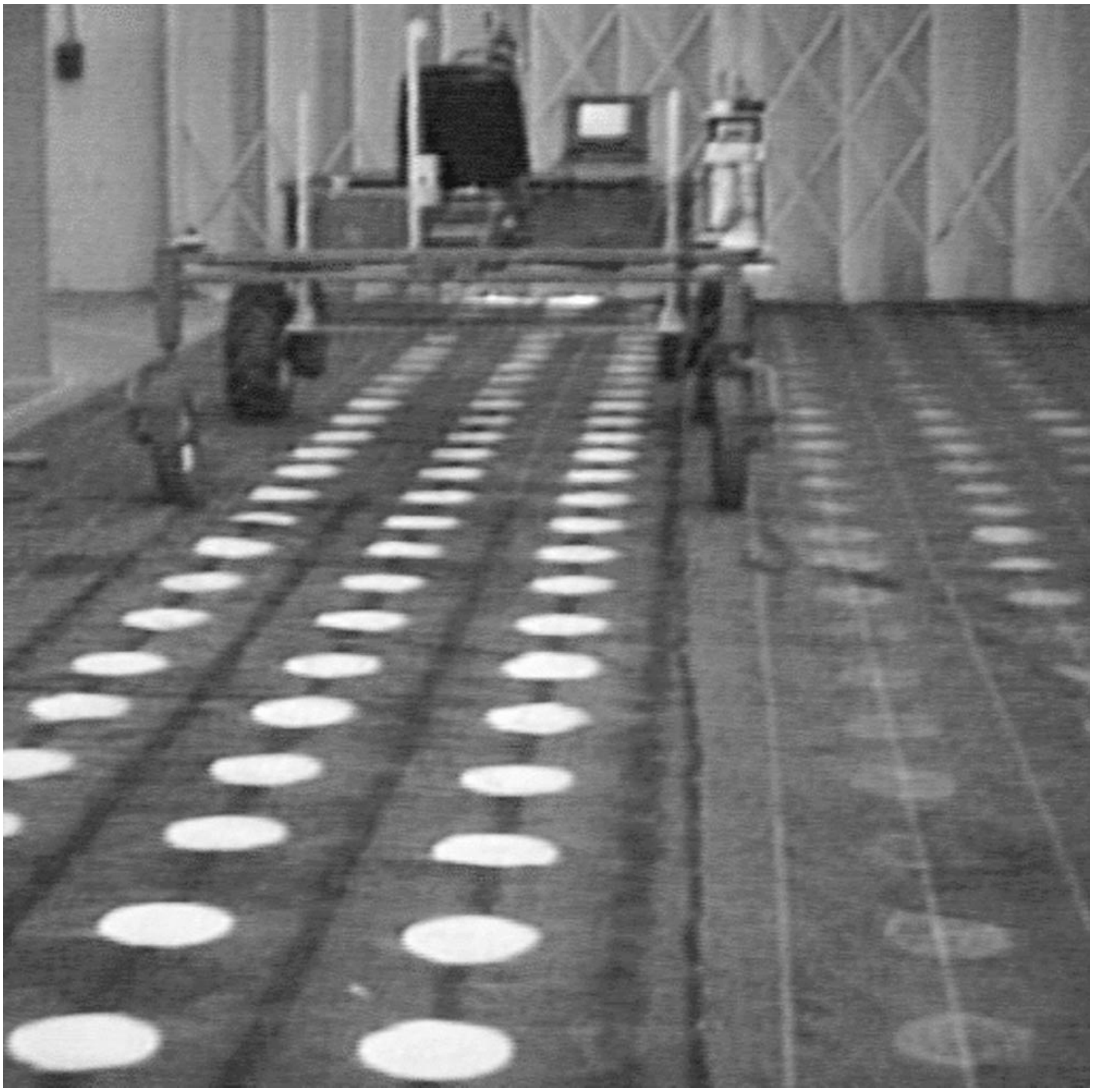


Figure 8. Autonomous vehicle in a full-size simulated field.

device in a field navigation vehicle; then to explore this map along the position of a nozzle bar and to switch on the nozzles that are over a plant (or weed), thus applying the treatment. A multi-processor architecture has been used, dedicating each processor to the implementation of one module of the system, and using serial links to pass images and data between processors. A method to process the images fitting the time requirements has been presented. Also, off-line and real-time results have been reported, including computational timing and accuracy measurements.

Acknowledgements

This work has been partially supported by the BBSRC (UK) and Fundació Caixa Castelló (Spain).

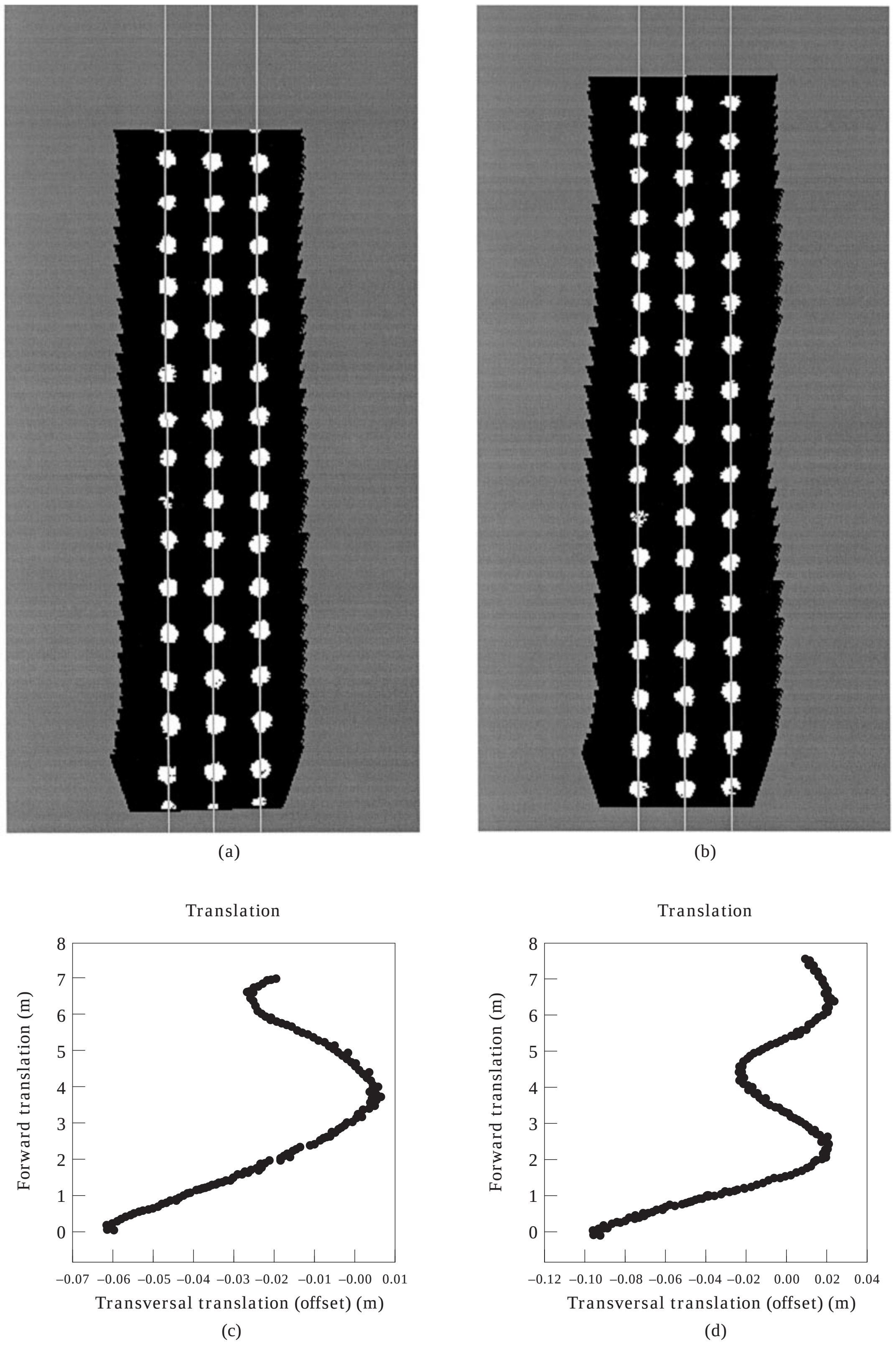


Figure 9. (a, b) Recovered local maps for two real-time tests. (c, d) Vehicle trajectories.

References

1. Liou, S. H. & Jain, R. C. (1986) Road following using vanishing points. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 41–46.*
2. *Thomas, B. T., Dagless, E. L., Lotufo, R. A., Milford, D. I., Morgan, A. D. & Morrissey, J. F. (1988) Feature extraction for vision guided road vehicles. Proceedings of the 4th Alvey Vision Conference*, pp. 173–178.
3. Morgan, A. D., Dagless, E. L., Milford, D. J. & Thomas, B. T. (1988) Road edge tracking for robot road following. *Proceedings of the 4th Alvey Vision Conference*, pp. 179–184.
4. Lotufo, R. A., Dagless, E. L., Milford, D. J. & Thomas, B. T. (1988) Road edge extraction using a plan-view image transformation. *Proceedings of the 4th Alvey Vision Conference*, pp. 185–190.
5. Wallace, R., Stentz, A., Thorpe, C. Moravec, H., Whittaker, W. & Kanade, T. (1985) First results in robot road following. *Proceedings of the 9th International Joint Conference on Artificial Intelligence, 2*: 1089–1095.
6. Crespi, B., Furlanello, C. & Stringa, L. (1993) Memory-based Navigation. *Proceedings of the 13th International Joint Conference on Artificial Intelligence*, **2**: 1654–1658.
7. Stelmasyzk, P. Ishiguro, H. & Tsuji, S. (1991) Mobile robot navigation by an active control of the vision system. *Proceedings of the 12th International Joint Conference on Artifical Intelligence*, **2**: 1241–1246.
8. Charnley, D. & Blisseu, R. (1988) Surface reconstruction from outdoor image sequences. *Proceedings of the 4th Alvey Vision Conference*, pp. 153–158.
9. Santos, J. & Senteiro, J. (1992) Generation of 3D dense depth maps by dynamic vision. *Proceedings of the British Machine Vision Conference*, pp. 129–138.
10. Marchant, J. A. & Tillet, R. D. (1994) Software and transputer system design for high speed grading of agricultural produce. *Mechatronics*, **4**: 281–293.
11. Illingworth, J. & Kittler, J. (1987) A survey of efficient Hough transform methods. *Proceedings of the 3rd Alvey Vision Conference*, pp. 319–326.
12. Marchant, J. A. & Brivot, R. (1995) Real time tracking of plant rows using a Hough transform. *Real Time Imaging*, **1**: 363–371.
13. Brivot, R. & Marchant, J. A. (1996) Segmentation of plants and weeds using infrared images. *Proceedings of the IEE Vision, Image and Signal Processing.*. **143**: 118–129.
14. Tsai, R. Y. (1986) An efficient and accurate camera calibration technique for 3D machine vision. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 364–374.